On the Meaning of Message Sequence Charts

Manfred Broy
Institut für Informatik
Technische Universität München

Topics: We discuss Message Sequence Charts (MSCs)

- as a technique to describe patterns of the interaction between the components of a interactive system
- as a technique to specify the behavior of the components of a system
- their systematic use in the software development process

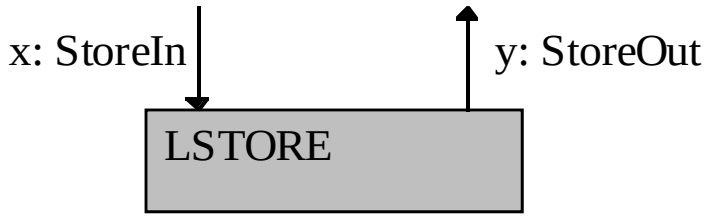
Questions:

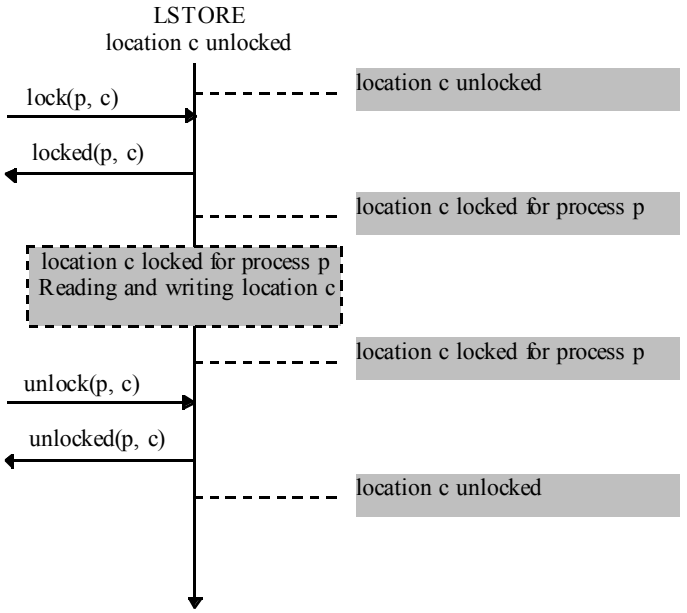
- What is the formal meaning of a MSC for the components of a system?
- How can MSCs be combined?
- What is the formal meaning of a set of MSCs?
- How far can MSCs serve as a precise and comprehensive mean of specification?
- What type of systems do MSCs refer to and what properties do they specify precisely?
- What is the methodological role of MSCs and how do they fit into the development process?

The Purpose of Message Sequence Charts

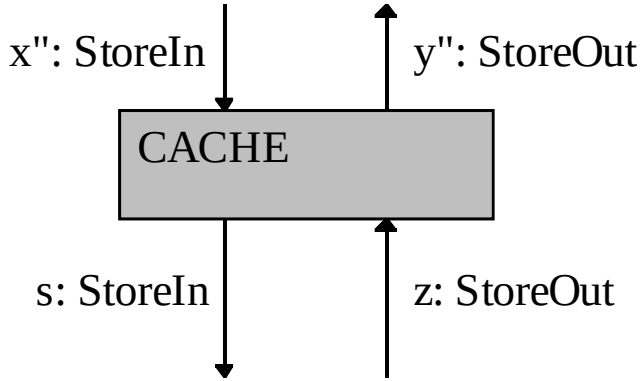
- MSCs as *logical properties of systems*.
- MSCs as *predicates* on system components.

Example: Component

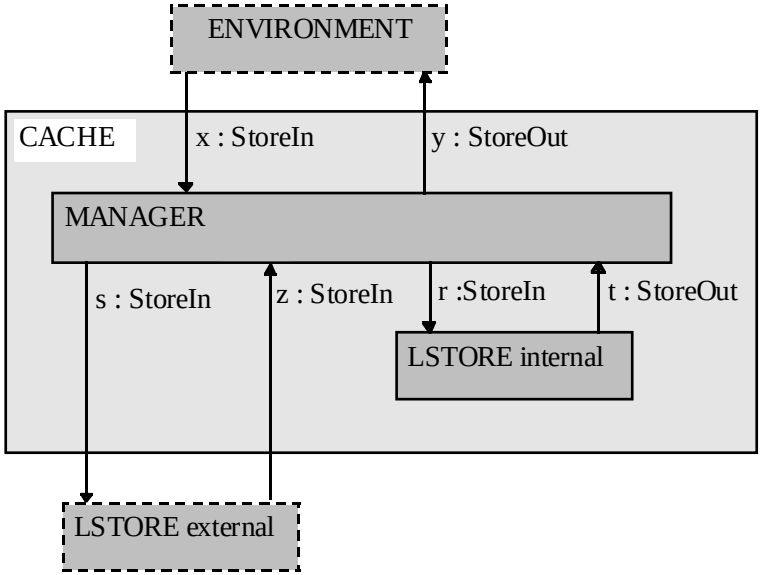




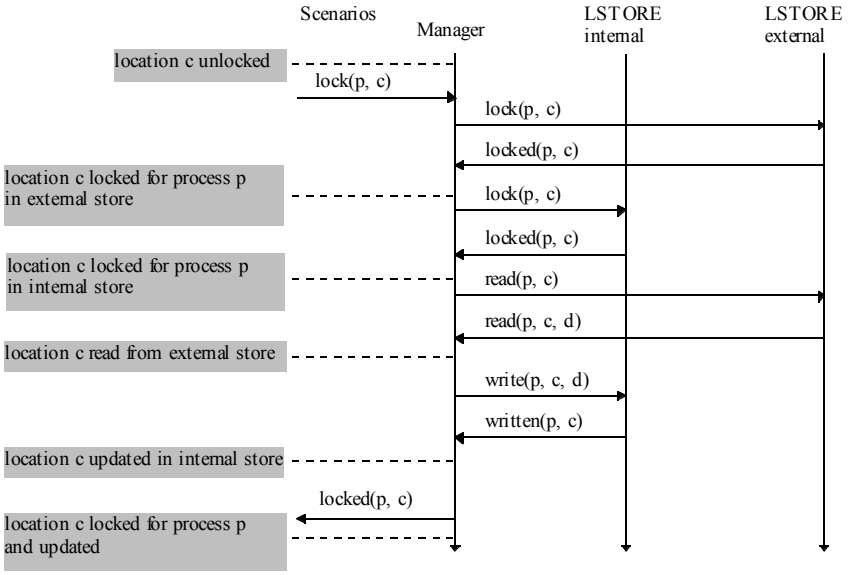
Successful Access to the Store



Cache as a Data Flow Node



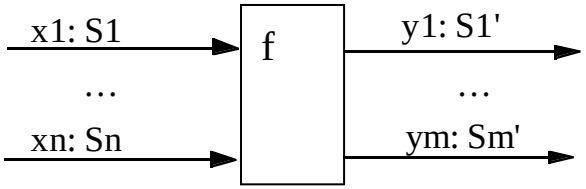
Cache as a Network of Data Flow Nodes



Message Sequence Chart for Successful Locking

Selected System Model

Components



I be the set of input channels

O be the set of output channels.

With every channel in the cannel set $I \cup O$ we associate a data type

(I, O) is the *syntactic interface* of a system component is given.

Streams

$M\omega = M^* \cup M^\infty$ streams of messages of set M

finite or infinite sequences of elements from M

$M\varpi$ set of infinite streams from set $M \cup \{\surd\}$ with an infinite number of time ticks $\surd$.

$M\varpi$ $\mathbb{N} \to M \cup \{\surd\}$.

Valuations of channels

C	set of channels
type: $C \to S$	types assigned to channels
$[\![s]\!]$	carrier set of data elements
$M = \bigcup\{[\![s]\!]: s \in S\}$	M be the universe of all messages
$x: C \to M\varpi$	valuations of C
	where for each $c \in C$: $x(c) \in [\![s]\!]\varpi$
$\vec{C}$	set of valuations

Black box behavior of a component

$f : \vec{I} \to \wp(\vec{O})$ I/O-functions

$x{\downarrow}k = z{\downarrow}k \Rightarrow \{y{\downarrow}k+1: y \in f(x)\} = \{y{\downarrow}k+1: y \in f(z)\}$ *timing property*.

$\bar{f}: (I \to M^\omega) \to \wp(O \to M^\omega)$ time abstraction

$\bar{f}.x = \{\bar{y} \in O \to M^\omega: \exists\, z \in \vec{I}: \bar{z} = x \wedge y \in f.z\}$

Com[I, O] set of I/O-functions with input channels I and output channels O.

Composed Systems by data flow nets

N set of identifiers for components

(ν, O) distributed system with syntactic interface (I, O)

$\nu: N \to Com$

Black box view: $f \in Com[I, O]$ where

$f(x) = \{y|_O: \; y|_I = x \wedge$

$\forall\, i \in N: y|_{Out(\nu(i))} \in \nu(i)(y|_{In(\nu(i))})\ )\ \}$

Towards a Meaning of MSCs

Given a network of interacting components with syntactic interface (I, O)

$(\nu: N \rightarrow Com, O)$

MSCs define

- a predicate $Q_p : \vec{I}_p \rightarrow \wp(\vec{O}_p)$

for each component $p \in N$ represented by a thread in the MSC.

Questions

(1) Is the component p described by the threads of the MSCs deterministic?

(2) Is the component, after a pattern of behavior as specified, in a state where again MSCs are available to describe the behavior?

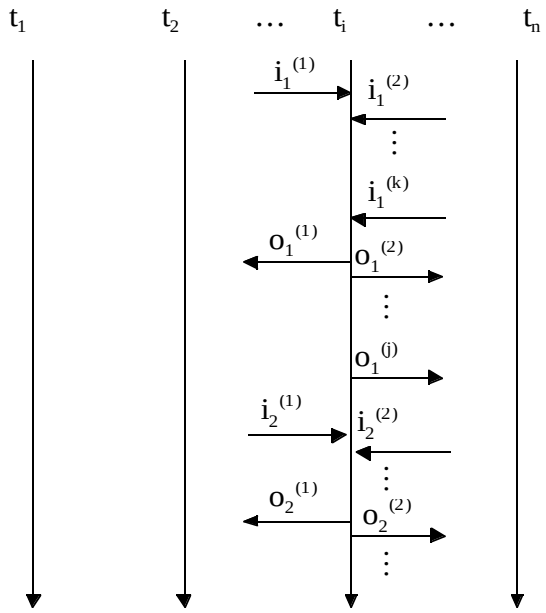
(3) Are the MSCs dense prefixes, projections, or free selections of instances of interactions?

(4) Is the set of MSCs meant as a loose sample or a comprehensive set of requirements?

Syntax and Structured Presentation of MSCs

A MSC is

- a diagram with n vertical lines, called the *threads*,
- one for each component of the system,
- that symbolize the flow of time for each component
- time flows from the top to the bottom.



Causal consistency of MSCs

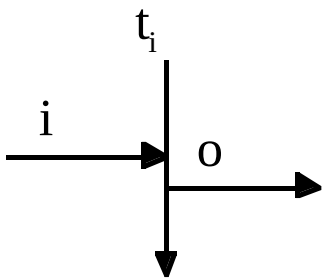
A MSC is called *consistent*, if for all its events there is

- a partial ordering that contains
- the linear orderings of all the threads as suborderings

(for a comprehensive discussion see [Alur et al. 96]).

This means:

- no cycles in the causality flow of the events (no "causal loops").



General Form of a Thread in a MSC with Clustered Input/Output

An input or output pattern is a finite channel valuation

$$\vec{C}^*$$

for the channels in C which is defined as a mapping

$$C \to (M \cup \{\surd\})^*.$$

Semantics of Sets of MSCs Specifying Deterministic Systems

Behavior of a deterministic system component

$$f: \vec{I} \to \vec{O}$$

The MSC is represented by the equation

$$\bar{f}(i) = o$$

The Semantics of MSCs for Nondeterministic Systems

A nondeterministic system component is interpreted by a set valued function

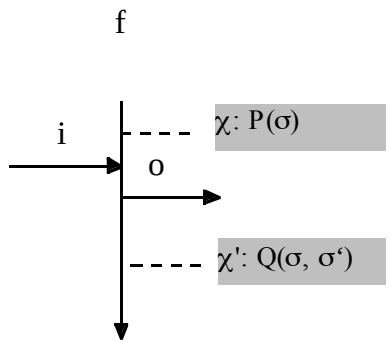
$$f: \vec{I} \to \wp(\vec{O})$$

and the specifying formula

$$o \in \bar{f}(i)$$

We call a function f *comprehensive* with respect to a set of MSCs if it fulfills the specifying formulas.

Introducing Control and Data States



$o \in \bar{f}^{\chi}_{\sigma}(i) \quad \Leftarrow P(\sigma) \quad \text{for } 1 \leq k < n$

$o \hat{} \bar{f}^{\chi'}_{\sigma'}(x) \subseteq \bar{f}^{\chi}_{\sigma}(i \hat{} x) \quad \Leftarrow P(\sigma) \wedge Q(\sigma, \sigma')$

This leads to a system of specifications of the function $\bar{f}^{\chi}_{\sigma}$.

The Universe of Unfiltered Scenarios

We get a set of conditional formulas of the general forms

$$\text{condition}(x, y) \Rightarrow y \in \bar{f}^{\chi}_{\sigma}(x)$$

or

$$\text{condition}(x, y) \Rightarrow y = \bar{f}^{\chi}_{\sigma}(x)$$

By choosing all instantiations for the input history x and the output history y

$$y \in \bar{f}^{\chi}_{\sigma}(x)$$

or

$$y = \bar{f}^{\chi}_{\sigma}(x)$$

for concrete histories $y \in \vec{O}$ and $x \in \vec{I}$.

MSCs as Projections

Given projection functions for a stream x written in the form

$$x \mid_N$$

We specify then a set of MSCs by the formulas

$$i = x \mid_N \Rightarrow \exists\, y \in \bar{f}(x): o = y \mid_N$$

interpretion of the threads of a component:

–*projection*: a MSC describes a prefix of a projection of the input and output,

–*loose selection*: a MSC describes a loose selection of messages.

This set of the patterns is called the *filtered universe of interaction patterns*.

Closed World Assumptions - A Canonical Behavior with a Set of MSCs

An I/O-function

$$f: \vec{I} \rightarrow \wp(\vec{O})$$

connects infinite input histories with infinite output histories.

We define the meaning of a set of threads for the function f by

$$f^*: \vec{I}^* \rightarrow \wp(\vec{O}^*)$$

where for a set of channels C we define C^* as the set of valuations

$$C \rightarrow (M^*)^* \cup (M^*)^\infty$$

prefix ordering $\sqsubseteq$

$$DCS(\vec{O}^*) = \{Z \subseteq \vec{C}^*: Z \neq \emptyset \wedge \forall z \in C^*: z \sqsubseteq x \wedge x \in Z \Rightarrow z \in Z\}$$

The partial ordering that we use on the set DCS is simply set inclusion.
We define the function f^* such that

$$f^*: \vec{I}^* \rightarrow DCS(\vec{O}^*)$$

For set of formulas $R = \{\Phi_k: k \in K\}$ where each Φ_k is of the form

$$y_k \in \bar{f}(x_k) \text{ or } \quad \bar{f}(x_k \hat{} x') \subseteq y_k \hat{} \bar{f}(x')$$

we specify the function $\bar{f}^*$ as the $\subseteq$-least time guarded function where

$$y_k \in \bar{f}^*(x_k)$$

$$\bar{f}^*(x_k \hat{} x') \subseteq y_k \hat{} \bar{f}^*(x')$$

Closures on the Canonical Behavior

The function f^* is completed into an I/O-function

$$f: \vec{I} \rightarrow \wp(\vec{O})$$

by the following equation:

$$\bar{f}.x = \{y \in \vec{O}: \quad \forall k \in \mathbb{N}: \overline{y \downarrow k} \in \bar{f}^*(x) \vee$$
$$\exists k \in \mathbb{N}: \overline{y \downarrow k} \in \bar{f}^*(x) \wedge \forall y' \in \bar{f}^*(x): \overline{y \downarrow k} \sqsubseteq y' \Rightarrow \overline{y \downarrow k} = y'\}$$

The construction of the *chaotic closure*

The function f is the inclusion largest function with the following properties:

(1) f is time guarded,

(2) if there is a MSC that talks about the input x, then f(x) contains exactly those outputs explicitly described in one of the MSCs.

(3) if for an input x we can decompose x into x'^x'' such that exactly for the input pattern in x' output is specified by the MSCs we define the output for x'' as being arbitrary.

In particular, whenever for some input an input pattern is not contained in a set of MSCs, the output can be arbitrary (chaos).

The Methodological Role of MSCs

(1) In the early phases of requirements engineering to get a first idea which services the system should provide.

(2) In the later phases of requirements engineering as description techniques as part of a formal specification.

(3) In the design phase illustration of the interaction between the system parts (key technique for the decomposition of a system, design patterns).

(4) In the implementation phase we use MSCs to represent test cases.

Poperty refinement

$\hat{f}: \vec{I} \rightarrow \wp(\vec{O})$

is called a *property refinement of a component* with a behavior

$f: \vec{I} \rightarrow \wp(\vec{O})$

if for all input streams $x \in \vec{I}$ we have: $\hat{f}(x) \subseteq f(x)$

Conflict between refinement and requirements capture by sets of MSCs:

refinement steps that allow us to get rid of nondeterministic alternatives,

the sets of MSCs that are assumed to describe all behaviors that all should be possible.

First answer [Klein 98]

Let us understand a set of scenarios as

- the specification of the components
- that shows for input patterns covered by scenarios exactly the behavior described by the scenarios and
- for the input patterns not covered arbitrary behavior.

See "closed world assumption".

Second Answer to the Conflict Refinement/Sets of MSCs

Distinguishing two kinds of MSCs

- Good MSCs: MSCs that describe the intended behavior and interaction,
- Bad MSCs: MSCs that describe unwanted behavior, modeling failure cases that, however, have to be tolerated, but should be avoided whenever possible.

Negative scenarios may be eliminated in refinement steps whenever feasible.

Conclusion

Several ways to interpret sets of MSCs -

Suggested concepts:

- distinction between input messages and output messages essential.
- leads to a concept of *causality*.
- MSCs as component specifications
- MSCs in that way do not only describe by an MSC very loosely what *may* happen in a system but also quite strictly what *must* happen.